AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS:

Please cancel claim 1 of the original parent application.

2. (new) A computer program product, disposed on a computer readable medium, the product including instructions for causing a processor to:

access Transmission Control Protocol (TCP) segments of a bidirectional TCP connection between a first TCP end-point operating at a first network device and a second TCP end-point operating at a second network device;

determine a first TCP state machine state of the first TCP end-point based on at least some of the accessed TCP segments;

determine a second TCP state machine state of the second TCP end-point based on at least some of the accessed TCP segments;

reassemble data from accessed TCP segments sent from the first TCP end-point to the second TCP end-point based on sequence numbers of the TCP segments, at least some of the TCP segments being received out of order; and

reassemble data from accessed TCP segments sent from the second TCP end-point to the first TCP end-point based on sequence numbers of the TCP segments, at least some of the TCP segments being received out of order.

3. (new) The computer program product of claim 2, wherein the instructions that determine the TCP state machine state of the first and second end-point comprise instructions that determine a change in the TCP state machine state of the first and second end-point.

4. (new) The computer program product of claim 2, wherein the instructions comprise instructions of a software library.

5. (new) The computer program product of claim 4, wherein the instructions comprise instructions of at least one object-oriented class.

6. (new) The computer program of claim 5, wherein the at least one object-oriented class comprises at least one of the following: a class for the bidirectional connection, a class for a TCP end-point, and a class for TCP segment reassembly.

7. (new) The computer program product of claim 2, wherein the instructions to reassemble data from accessed TCP segments sent from the first TCP end-point to the second TCP end-point comprise instructions that maintain a linked list, wherein the reassembled data is stored in discontiguous memory locations linked by the linked list.

8. (new) The computer program product of claim 2, wherein the instructions further comprise instructions to provide a return code at least one of: indicating whether a TCP segment was received out-of-order and indicating whether a TCP segment overlapped a another TCP segment.

9. (new) The computer program product of claim 2, wherein the instructions further comprise instructions to return data of a field within a header of a TCP segment.

10. (new) The computer program product of claim 2, further comprising application program instructions that invoke the instructions to access TCP segments, determine the first TCP state machine state, determine the second TCP state machine state, reassemble data from the accessed TCP segments sent from the first TCP end-

point to the second TCP end-point, and reassemble data from the accessed TCP segments sent from the second TCP end-point to the first TCP end-point.

11. (new) A method, comprising:

providing a software library featuring operations to:

access Transmission Control Protocol (TCP) segments of a bidirectional TCP connection between a first TCP end-point operating at a first network device and a second TCP end-point operating at a second network device;

determine a first TCP state machine state of the first TCP end-point based on at least some of the accessed TCP segments;

determine a second TCP state machine state of the second TCP end-point based on at least some of the accessed TCP segments;

reassemble data from accessed TCP segments sent from the first TCP end-point to the second TCP end-point based on sequence numbers of the TCP segments, at least some of the TCP segments being received out of order; and

reassemble data from accessed TCP segments sent from the second TCP end-point to the first TCP end-point based on sequence numbers of the TCP segments, at least some of the TCP segments being received out of order.

12. (new) The method of claim 11, wherein the determining the TCP state machine state of the first and second end-point comprises determining a change in the TCP state machine state of the first and second end-point.

13. (new) The method of claim 11, wherein the software library comprises at least one object-oriented class.

14. (new) The method of claim 11, wherein the at least one object-oriented class comprises at least one of the following: a class for the bidirectional connection, a class for a TCP end-point, and a class for TCP segment reassembly.

15. (new) The method of claim 11, wherein the operations to reassemble data from accessed TCP segments sent from the first TCP end-point to the second TCP end-point comprise operations that maintain a linked list, wherein the reassembled data is stored in discontiguous memory locations linked by the linked list.

16. (new) The method of claim 11, wherein the operations further comprise operations to provide a return code indicating at least one of: whether a TCP segment was received out-of-order and whether a TCP segment overlapped another TCP segment.

17. (new) The method of claim 11, wherein the operations further comprise operations to return data of a field within a header of a TCP segment.

18. (new) The method of claim 11, further comprising invoking the software library operations from an application program.